

Position Paper on xrootd Software for Exascale Systems

A. Hanushevsky, R. Mount, W. Yang

SLAC National Accelerator Laboratory, July 13, 2012

Contact: Richard P. Mount, richard.mount@slac.stanford.edu

Introduction

Xrootd is a robust, high-performance, component software system supporting request/response services such as access to multi-petabyte data and highly concurrent database services.

Xrootd was conceived to support scalable data access for particle physics experiments, meeting a need for high performance and much greater robustness than any of the underlying servers. Xrootd is now in use in diverse environments ranging from the worldwide federation of data from the Large Hadron Collider, to the high transaction rates of the petabyte-scale catalog database designed for the Large Synoptic Survey Telescope.

Xrootd has not yet been applied to exascale systems. We believe that it deserves serious examination and experimentation in an exascale context. We propose research – likely an iterative series of prototype deployments followed by further R&D – to attempt to establish xrootd as part of the run-time toolkit empowering exascale computing.

Challenges addressed: Which exascale OS/R challenges does this approach address?

Data-intensive computation will grow in importance for exascale systems. The traditional strengths of xrootd – efficient, robust access to data residing on tens to thousands of servers – will certainly be applicable. More generally, xrootd has shown that it can provide a low-latency request/response switchyard function linking tens of thousands of compute cores. Xrootd assumes no master intelligence, no reliable catalogs, and that any component of an exascale system may become unavailable. It thus allows extreme scaling with no inherent architectural bottlenecks.

Maturity: What are the indicators that this approach will address the identified challenges?

Xrootd is successful in its current applications, and would almost certainly bring similar benefits to data-access-intensive exascale computing. Xrootd provides request-for-function brokerage for the LSST object catalog. We believe that this type of capability will be important for future exascale problems.

Uniqueness: To what extent is the proposed approach unique to exascale systems? Could it be addressed by other research programs?

The xrootd architecture is widely applicable rather than unique to exascale systems. However, different aspects of the architecture become stressed in its various extreme areas of application. We believe that it is likely that while exascale systems may drive the development of substantial new or improved software within xrootd, that this can be accomplished while maintaining architectural coherence. That said, the unique feature of xrootd is that it has no inherent limit to the number of nodes that it can support. Indeed it was architected to naturally scale to meet exascale demands.

Other research programs will certainly address many of the functional goals of xrootd, but the maturity of the xrootd approach argues strongly for an exploration of its applicability to exascale systems.

Novelty: How is this approach different than existing solutions?

Xrootd's strengths are in the data-intensive aspects of computing, in supporting massively concurrent asynchronous communications, in invoking services by function or name without a need to know physical address, and generally not getting upset when what worked a minute ago no longer does (a.k.a. redirection technology). The communication framework also allows for dynamic partitioning of arbitrary end-points. This feature is significant and distinct in the context of how exascale systems are likely to be used in practice. We believe that the xrootd approach has only a limited overlap with current exascale run-time solutions.

Applicability: To what extent will the proposed approach, if successful, be applicable to other areas?

To some extent this is already demonstrated, since xrootd is an established success in other extreme, but not strictly exascale, areas of scientific computing. Experience to date has shown that the xrootd architecture and core code benefits greatly from application to new extremely demanding challenges. To date, core development has not been forked to address widely different applications – every application that heavily stresses xrootd in one area drives improvements that have no downsides.

Effort: How much effort is needed to effectively explore this approach?

Xrootd is part of the run-time environment. In exploratory use it is often wrapped (for example to make its data-access function look like a POSIX compliant filesystem), but for the work at the performance frontier its protocol driver must be integrated into application software (or application-support software). The effort required for an effective exploration is thus the combination of the xrootd development effort, the application adaptation effort, and the at-scale deployment and benchmarking effort. A team of three FTEs, each with prime responsibilities in these three effort areas, would be able to deliver concrete results within two years.